Joint prediction of observations and states in time-series based on belief functions

Emmanuel Ramasso, Michèle Rombaut, Noureddine Zerhouni

Joint prediction of observations and states in time-series: a partially supervised prognostics approach based on belief functions and KNN

Emmanuel Ramasso and Michèle Rombaut and Noureddine Zerhouni
FEMTO-ST Institute, UMR CNRS 6174 - UFC / ENSMM / UTBM,
Automatic Control and Micro-Mechatronic Systems Department, 25000, Besançon, France
emmanuel.ramasso@femto-st.fr, noureddine.zerhouni@ens2m.fr
GIPSA-lab, UMR CNRS 5216 - UJF,
Signal and Images Department, 38000 Grenoble, France
michele.rombaut@gipsa-lab.inpg.fr

***Abstract*—Forecasting the future states of a complex system is a complicated challenge that is encountered in many industrial applications covered in the community of Prognostics and Health Management (PHM). Practically, states can be either continuous or discrete: Continuous states generally represent the value of a signal while discrete states generally depict functioning modes reflecting the current degradation. For each case, specific techniques exist. In this paper, we propose an approach based on case-based reasoning that jointly estimates the future values of the continuous signal and the future discrete modes. The main characteristics of the proposed approach are the following: 1) It relies on the K-nearest neighbours algorithm based on belief functions theory; 2) Belief functions allow the user to represent his partial knowledge concerning the possible states in the training dataset, in particular concerning transitions between functioning modes which are imprecisely known; 3) Two distinct strategies are proposed for states prediction and the fusion of both strategies is also considered. Two real datasets were used in order to assess the performance in estimating future break-down of a real system.**

I. Introduction

A. Problem statement

Forecasting the future states of a complex system is a complicated challenge that arised in many industrial applications covered in the community of Prognostics and Health Management (PHM) such as locomotive's health prediction [1], analysis of fleet of vehicles [2] and turbofan engine monitoring [3].

Practically, states can be either continuous or discrete:

- Continuous states generally represent the value of a signal (an observation or a feature) and its prediction can be made by Kalman-like procedures or by neural networks [4], [5],
- Discrete states generally depict functioning modes reflecting the current degradation and its prediction can be performed by state machines such as Hidden Markov Models [21].

In both cases, data-driven prognostics generally involves a training procedure where statistical models of the degradation are built. However, in real applications, one is facing lack of knowledge on the system since many unknown factors can not be identified, for example environmental conditions and particular functioning modes. These factors may play a crucial role in the data collection process and therefore in the degradation modelling, limiting the applicability of statistical models. In [7], this problem is underlined and tackled in the context of neuro-fuzzy systems.

To cope with the problem of lack of knowledge, case-based reasoning (CBR) was proposed as an efficient alternative to perform prognostics. For example, the method described in [3] demonstrated better performance than neural network for continuous state prediction in a turbofan engine. For that, historical instances of the system - with condition data and known failure time - are used to create a library of degradation models. Then, for a test instance of the same system, the similarity between it and the degradation models is evaluated generating a set of *Remaining Useful Life* (RUL) estimates which are finally aggregated by a density estimation method. Note that Section II of this paper is dedicated to the presentation of CBR approaches.

The main problem with the approach described in [3] is the number of parameters that has to be estimated in order to apply it. Moreover, several parts of the algorithm rely on statistical learning procedures requiring large amount of data.

The proposed EVIPRO-KNN algorithm requires a training dataset composed of trajectories (historical information). It takes as input an observation (test instance) in the form of a piece of a trajectory and then generates the prediction as the weighted sum of the K-nearest trajectories found in the training dataset. This procedure is made online (as data arrive) and is very common in most of KNN-based prognostics.

B. Contribution

Compared to previous KNN-based prognostics approaches (see [3] and references therein), EVIPRO-KNN requires less parameters and has the following characteristics:

1) *EVIPRO-KNN is a new prognostics approach based on belief functions:* A trajectory similarity-based approach

based on belief functions is proposed for prognostics. Belief functions were justly proposed to cope with lack of data in data representation, combination and decision-making [8]–[11] and account for both variability and incomplete knowledge by drawing benefits from both probability theory and set-membership approaches in one common and sound framework.

2) *EVIPRO-KNN takes into account partial labelling on states:* In some applications, the training dataset is composed of continuous trajectories and of a set of labels reflecting the current system state. These labels can be obtained by a manual annotation as it is commonly done in supervised classification [4], by a clustering method [12] or by a posteriori classification [6]. If these labels are known only partially, then belief functions can be used [13].
3) *EVIPRO-KNN manages trajectories with different temporal length:* The weighted sum of trajectories used to compute the prediction of observations requires trajectories with the same length, that is generally false in most of the applications. As far as we know, this problem of length was not treated in the past while it can have strong influence on the final result. We described two approaches to solve it.
4) *EVIPRO-KNN is able to predict jointly continuous and discrete states:* The prediction of the future sequence of states is performed jointly with the prediction of continuous observations. As far as we know, the joint prediction of discrete states and of continuous observations was not considered jointly in PHM applications nor in CBR-based prediction.

The possibility to manage prior information on possible states in the training dataset is one of the main advantages of EVIPRO-KNN. Indeed, in most papers, only two states are considered: normal and faulty. But in many real cases, more states have to be considered. When knowledge on states is not always certain and precise then belief functions can be used as described in this paper.

The other main asset of EVIPRO-KNN is the possibility to predict sequence of continuous observations jointly with discrete states. These sequences allow the user to have access to the online segmentation of the current observed data and may be practically useful. As shown in experiment, sequences of states generate accurate estimate of the Remaining Useful Life (RUL) of the system.

Finally, RUL estimation is generally based on the study of an one-dimensional degradation signal: If this signal becomes greater than a given threshold then the system is said to enter in a potential dangerous mode. This *system's health assessment* [14] requires the threshold to be tuned precisely and this can be a pratical problem, in particular when the signal does not have a physical meaning. Moreover, the use of multi-dimensional degradation signals is prefered to ensure reliable RUL estimates making the use of thresholding techniques practically difficult. In comparision, the proposed EVIPRO-KNN algorithm is based on a classification process which enables one to assess the *discrete* state (functioning mode) of the system while allowing the use of multi-dimensional degradation signal [6].

The remainder of this paper is organized as follows. Section II presents the related work. Section III is dedicated to the description of the basics of belief functions and of the notations. The proposed algorithm is described in Section IV. Finally, Section V demonstrates the performance of the approach on real data.

II. Related work

In Prognostics and Health Management (PHM) applications, approaches generally aim at computing long-term predictions of continuous observations followed by a thresholding in order to detect the fault mode and to, finally, estimate the Remaining Useful Life (RUL) of the system. Long-term prediction of continuous observations can be made by several data-driven techniques such as neural networks and statistical models. These approaches require a significative amount of representative data and generally generates black boxes. However, in PHM applications, the lack of data is an important problem in particular for faulty states which are difficult to obtain without degrading the system. The cost to obtain these data may be important and solutions have thus to be proposed. K-nearest neighbours-based approaches, which is a well known non-parametric solution for pattern analysis [4], was shown to be adapted because it fully exploits the training dataset and is adapted when the latter is small. For example in [3], KNN demonstrated high performance better than the previous techniques with lower complexity and better interpretation.

For prognostics applications [1]–[3], the estimation of the Remaining Useful Life (RUL) by KNN-based approaches generally involves three tasks [3]: instance retrieval in the training dataset (as in the usual KNN), prediction through local models and aggregation of local predictions. The actual life of the training instance is generally used as the prediction of the test instance's life. The local predictions are aggregated to obtain the final RUL estimate using weighted sum of the local predictions. Recent work such as [3], [15] are focused on associating a confidence information to predictions made by KNN, but solutions generally rely on density estimation or require representative prior information in the training dataset.

One can note that KNN-based systems for prediction were proposed in 90's, such as [16], where an empirical and comparative studies are made between Case-Based Reasoning (CBR) and other alternative approaches. Several applications in time-series forecasting were also developed based on KNN. For example in telecom industry, customer churn prediction represents a key priority. In [17], D. Ruta *et al.* addressed the weakness of static churn prediction and propose new temporal churn prediction system. It uses K-nearest sequence (kNS) algorithm that learns from the whole available customer data path and is capable to generate future data sequences along with precisely timed predicted churn events. Many applications concern finance, for example with [18] where KNN are used to build a prediction model of the return on assets of a company, in [19] where ensemble of KNNs are exploited for bankruptcy prediction, and in [20] where KNN is used for forecasting the final price of an ongoing online auction.

Recent approaches in PHM applications focused on long-term prediction of state sequences. In particular, in [21], statistical models of state duration based on Hidden Markov Models were exploited. However, this kind of solution is not always adapted since the duration can vary a lot according to the operational conditions. The use of states can avoid using thresholding techniques of continuous observations as proposed in [6] where an evidential markovian classifier was developed for the classification of predictions performed by a neuro-fuzzy system.

Describing the problem of PHM by using continuous states and discrete states is recent. In the community of automatic control, one can refer to hybrid systems which represent dynamical systems that exhibits both continuous and discrete behaviors. These systems are analyzed by several techniques such as switching models [22]. In the community of PHM, hybrid systems are generally understood differently since they refer to the combination of data-driven and physics-of-failure approaches [23] to improve long-term prediction.

Compared to existing work, this paper is focused on prognostics based on a small amount of data with high variability by using belief functions. EVIPRO-KNN algorithm is able to provide both predictions of states (discrete values) and of observations (generally continuous values), while in most of papers on PHM, methods are generally set for state prediction only or observations prediction only. We refer to this feature as *Joint Prediction of Observation and State (JPOS)*.

III. Background

Before describing the algorithm, belief functions are first presented followed by the formalization of the training data.

At each time t, an observation vector X_t can be extracted from the observed system. This system can be in one of the possible discrete states ω belonging to a set of S exhaustive and exclusive states $\Omega = \{\omega_1, \ldots, \omega_S\}$. The states can be imprecise and uncertain due to *aleatory uncertainty* induced by the variability in observations and to *epistemic uncertainty* induced by lack of knowledge [9]. For that, we describe the knowledge of states at time t by a belief function.

A. Belief functions

1) History: Belief functions were developed by Dempster [8], Shafer [9] and Smets [10] in the so-called Theory of Evidence and Transferable Belief Model. They combine probability and sets to account for both variability and incomplete knowledge. The theory of belief functions includes extensions of probabilistic notions such as conditioning and marginalization, and set-theoretic notions such as intersection, union, inclusion and so on. This theory was used successfully in several pattern recognition applications with static data where the main problem was to cope with the lack of data [24]. Belief functions applied to dynamical data is more recent and was shown to be promising for data stream processing [12], [25].

2) Representation: The basis in the theory of belief functions is the basic belief assignment (BBA) defined on a frame of discernment Ω by:

$$\begin{array}{rccc} m_t & : & 2^\Omega & \rightarrow & [0,1] \\ & & S & \mapsto & m_t(S) \end{array} \tag{1}$$

with $\sum_{A\subseteq\Omega} m_t(A) = 1$. The belief mass $m_t(A)$ represents the uncertainty (since $m_t(A) \in [0,1]$) and imprecision (since A is a subset with cardinality $|A| \geq 1$) about the possible state of the system at time t. Subset A is composed of unions of singletons ($\omega \in \Omega$) and thus represents explicitly the *doubt* concerning the value of the state. If the state is precisely known at time t, say ω, then the whole mass is assigned to ω, i.e. $m_t(\omega) = 1$. On the contrary, if the state is fully unknown at time t then the whole mass is assigned to the ignorance, i.e. $m_t(\Omega) = 1$. In the latter case, the BBA is called *vacuous*.

3) Combination: Given two BBA, say m_1 and m_2 defined on the same frame Ω, one may be interested in drawing benefits of both BBA to improve decision-making. For that, if the BBAs are generated by *distinct* bodies of evidence [26], the classical Dempster's rule can be used [27]:

$$m_{12}(C) = \sum_{A\cap B=C} m_1(A)\cdot m_2(B) \tag{2}$$

The mass on conflict given by $m_{12}(\emptyset)$ can be cancelled out by Dempster's normalisation consisting in dividing all masses to the opposite of conflict ($m_{12}(C)/(1-m_{12}(\emptyset))$). Other rules have been proposed [28]. Note that the vacuous BBA is the neutral element of Dempster's rule, i.e. combining any BBA m with the vacuous BBA provides m.

Two BBAs coming from *non-distinct* bodies of evidence [26] can be combined using the cautious rule. This rule is defined as:

$$w_{12}(C) = w_1(C) \wedge w_2(C) \tag{3}$$

where $\wedge$ is the *minimum* and

$$w(C) = \prod_{C\subseteq B} q(B)^{(-1)^{|B|-|C|+1}} \tag{4}$$

and

$$q(B) = \sum_{B\subseteq A} m(A) \tag{5}$$

After fusion, the function w can be converted into a mass function m using inverse formula:

$$q(A) = \frac{\prod_{B\subseteq\Omega} w(B)}{\exp(\sum_{A\subseteq D} \log w(D))} \tag{6}$$

and

$$m(A) = \sum_{A\subseteq B} (-1)^{|B|-|A|} q(B) \tag{7}$$

4) Decision-making: From a belief mass (resulting from a fusion process), one may be interested in selecting the best singleton. For that, the pignistic transformation [10] can be used which computes a probability distribution on singletons from which the decision can be made:

$$\hat{\omega}_t = \underset{\omega\in\Omega}{\operatorname{argmax}} \sum_{A\subseteq\Omega, \omega\in A} \frac{m_t(A)}{|A|} \tag{8}$$

5) Implementation issues: One may use the Fast Moebius Transforms (FMT) to compute these quantities efficiently [29]. The package TBMLAB, available at http://iridia.ulb.ac.be/~psmets/, implements these functions on MATLAB.

B. The training dataset

The training dataset is denoted

$$\mathcal{L} = \{T_i\}_{i=1}^{N} \tag{9}$$

and is composed of N trajectories T_i defined by both a sequence of Q-dimensional observation vectors $X_t \in \Re^Q$ and the knowledge on the possible states:

$$T_i = \{(X_t^i, m_t^i)\}_{t=t_i}^{t_i+|T_i|} \tag{10}$$

For one trajectory, the knowledge on states is represented by a set of belief functions (BBAs) at each time-step. According to the priors available, these BBAs can take various forms which will be detailed further (next section).

The i-th trajectory begins at time t_i and finishes at time $t_i + |T_i|$ where $|T_i|$ is the length of T_i. To each trajectory T_i is associated a set of blocks $\mathbb{B}_i$ where each block B_i^j in this set corresponds to a sub-trajectory of length W:

$$B_j^i = \{(X_t^i, m_t^i)\}_{t=c_j}^{c_j+W} \tag{11}$$

where $c_j \in [t_i, (t_i + |T_i| - W)]$ is the starting time of the j-th block. The number of blocks (and thus the range of index j) in the i-th trajectory depends on the length of the latter. An illustration of blocks is given in Figure 1. Note that the blocks may overlap. In the sequel, the overlapping was set to $W/2$.

A trajectory T_i is thus composed of $|T_i|$ observation vectors and $|T_i|$ mass functions m_t defined on the frame Ω composed of the states in which the system can be.

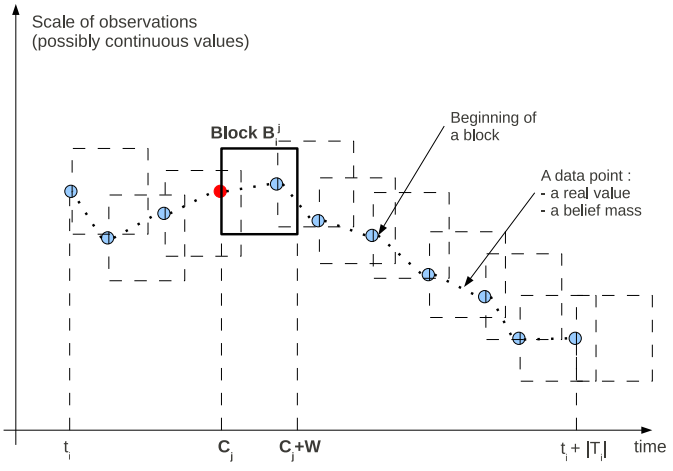


Figure 1. Illustration of blocks in trajectory T_i.

C. Partially supervised training

In some applications, the training dataset is composed of features and of a set of labels reflecting the current system's state. These labels can be obtained by a manual annotation as it is commonly done in supervised classification [4], by a clustering method [12] or by a posteriori classification [6].

If the labels are known only partially, then belief functions can be used [13]. Managing belief functions in the training dataset allows the user to consider three main cases:

- Supervised learning: the true state is perfectly known for all instances in the training dataset, i.e. $\forall i = 1 \dots N, m_t^i(\omega) = 1, \omega \in \Omega$.
- Semi-supervised learning: the true state is perfectly known for some instances only and totally unknown for the other instances. In the latter case, a vacuous belief mass (with full mass on ignorance) is set for these instances, i.e. $\forall i = 1 \dots N, m_t^i(\Omega) = 1$.
- Partially-supervised learning: the state is known with uncertainty and imprecision and thus described by a belief mass, i.e. $\forall i = 1 \dots N, m_t^i(A) > 0$ for some subsets $A \subseteq \Omega$.

Therefore, the *partial labeling* by using the belief functions formalism is a way to encode general situations of knowledge.

IV. EVIPRO-KNN ALGORITHM

Let now consider that a block of data $Y_t \in \Re^Q$ of length W is measured on the system. Given the training dataset and this observation, the goal is to predict an observation trajectory $\hat{T}_t = \{(\hat{X}_{t'}, \hat{m}_{t'})\}_{t'=t}^{t+H}$ where H is an horizon of prediction. The value of H will be set automatically as shown in the sequel. Note that the final algorithm is given in Alg. 1, and the plot chart of the whole algorithm is provided in Figure 6.

A. Step 1 - K-best trajectories determination

In this step, the K nearest trajectories to observations Y_t are determined. Note that all distances $\mathcal{D}$ are measured using the Euclidean distance.

For that, all trajectories in the training dataset $\mathcal{L}$ are scanned. For each trajectory T_i, the nearest block $B_{j^*}^i \in \mathbb{B}_i$ to the observation block Y_t is found. Index j^* of the best block $B_{j^*}^i$ in the i-th trajectory is given by:

$$j^* = \underset{j, B_j^i \in \mathbb{B}_i}{\operatorname{argmin}} \ \mathcal{D}(Y_t, B_j^i) \tag{12}$$

When the best block in each trajectory has been found, all best blocks are sorted by ascending order according to their distance:

$$\mathcal{D}_{j^*}^i \equiv \mathcal{D}(Y_t, B_{j^*}^i) \tag{13}$$

Let $\mathcal{D}_{j^*}^{(i)}$ denote one element of this partial ordering with $\mathcal{D}_{j^*}^{(1)} \leq \mathcal{D}_{j^*}^{(2)} \leq \dots \mathcal{D}_{j^*}^{(i)} \leq \dots \mathcal{D}_{j^*}^{(N)}$. Finally, the K best trajectories $T_k, k = 1 \dots K$ are simply the ones associated to the K best and sorted blocks:

$$\mathcal{D}_{j^*}^{(1)} \leq \mathcal{D}_{j^*}^{(2)} \leq \dots \mathcal{D}_{j^*}^{(k)} \leq \dots \mathcal{D}_{j^*}^{(K)} \tag{14}$$

The K selected trajectories $T_k = \{(X_t^k, m_t^k)\}_{t=c_k}^{|T_k|}, k = 1 \dots K$ are composed of both a set of features $X_t \in \Re^Q$ and knowledge m_t about the state. The index c_k is known by means of Eq. 12 and represents the starting time of the best block $B_{j^*}^k$ in the k-th best trajectory.

Step 1 is represented in lines 1-6 of Alg. 1.

The next steps of the algorithm consists in aggregating trajectories $T_k, k = 1 \dots K$ where two problems arised:

- How to aggregate the features $\{X_t^k\}_{t=c_k}^{|T_k|}, k = 1 \dots K$ to obtain a predicted set of features $\hat{X}_t$? (Step 2)
- How to aggregate the knowledge about states $\{m_t^k\}_{t=c_k}^{|T_k|}, k = 1 \dots K$ to obtain a predicted knowledge $\hat{m}_t$? (Step 3)

B. Step 2 - Predicted observation trajectory

A simple and usual way (common in KNN-based approaches) to define a predicted observation trajectory $\hat{X}_t$ linked to the observation block Y_t is to compute the weighted average of the K sets of features:

$$\hat{X}_{t+h} = \sum_{k=1}^{K} F^k \cdot X_l^k, l = c_k \dots |T_k|, h = 1 \dots \mathcal{P} \qquad (15)$$

where

$$\mathcal{P} = |T_k| - c_k + 1 \qquad (16)$$

defines the set of instants of prediction. The normalized weights F^k are obtained by the softmax function of the sorted distances (Eq. 13):

$$F^k = \frac{\exp\left(-\mathcal{D}_{j^*}^{(k)}\right)}{\sum_{k'=1}^{K} \exp\left(-\mathcal{D}_{j^*}^{(k')}\right)}, k = 1 \dots K \qquad (17)$$

The use of the softmax transformation is interesting for several reasons: it generates (as expected) weights which decrease as the distances (potentially not bounded) increase, and the exponential ensures a better constrast between the generated values. The softmax transformation was used in several applications, in particular with classifiers such as Support Vector Machines and Neural Networks in order to generate (posterior) probability distributions [30].

An illustration of the prediction process for observations is given in Fig. 2: given a block, the prediction is computed by Eq. 15, and the horizon depends on the length of the prediction. Each point of the prediction is a real value representing an *observation*.

For $K > 1$, equations 15 and 17 are directly used if the length of trajectories $T_k, k = 1 \dots K$ is identical. If it is not the case (and generally it is not), one can use two different strategies: *cautious* and *bold*[1]. The former consists in truncating the trajectories to the length of the shortest one, while the latter keeps all trajectories as such. These process are depicted in Figure 3, and are described in the sequel.

1) Cautious strategy: This strategy consists in selecting an horizon of prediction equal to the length of the smallest trajectory. For that, first, the trajectory with the smallest size is found:

$$H_t = \min_{k=1}^{K} |T_k| \qquad (18)$$

where H_t is the horizon of prediction at time t. Then, for all trajectories, only samples from c_k to H_t are kept. After removal of samples located beyond H_t, Equations 15 and 17 can be directly used:

$$\hat{X}_{t+h}^{CS} = \sum_{k=1}^{K} F^k \cdot X_l^k, l = c_k \dots H_t, h = 1 \dots H_t \qquad (19)$$

where CS stands for "Cautious Strategy" and X_h^k is the value of features in trajectory T_k taken at time h. The value of F^k is given by Eq. 17.

The main advantage of this strategy is its simplicity and also its efficiency because, the horizon being shortened (to the smallest trajectory), it generally provides more reliable predictions. The main drawback is that the horizon of prediction is justly made shorter and therefore reducing forecasting capability.

2) Bold strategy: Let consider that the k-th best trajectory T_k starts from c_k (time instant corresponding to the beginning of the best block in T_k). Therefore, trajectories in $\mathcal{L}$ are truncated (from c_k) before computing the prediction and thus, length of trajectories $|T_{(k)}|$ is reduced to $|T_{(k)}| - c_k + 1$. Trajectories (truncated) are then sorted according to their length: $|T_{(1)}| < |T_{(2)}| < \dots |T_{(k)}| < \dots < |T_{(K)}|$. The

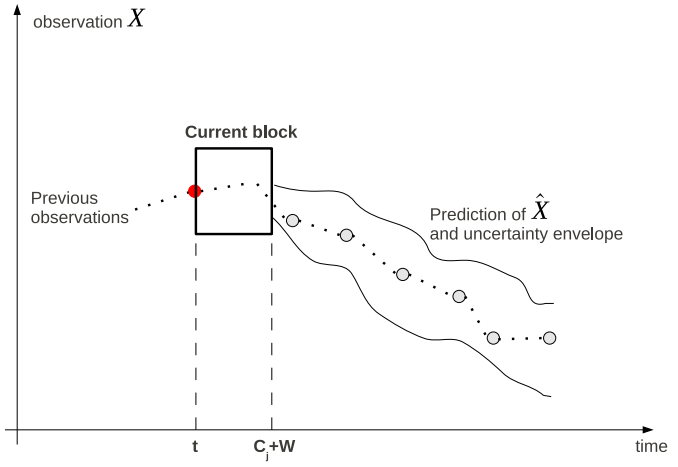


Figure 2. Illustration of the observation prediction process. Each data point is a value in the space of reals.

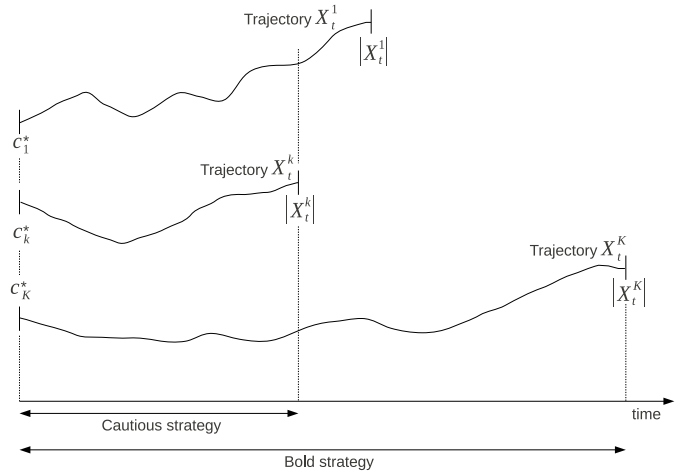


Figure 3. Illustration of the bold and cautious strategies.

[1]We get inspired from the work of T. Denoeux on combination rules of belief functions [26] to choose these terms.

average (similar process to Eq. 15 but adapted for the bold strategy in Eq. 20) is then taken on the K trajectories from their beginning to $|T_{(1)}|, \forall k = 1 \dots K$, then it is taken on $K-1$ trajectories $(T_{(2)} \dots T_{(k)} \dots T_{(K)})$ from $|T_{(1)}|+1$ to $|T_{(2)}|, \forall k = 2 \dots K$, and so on, until the last trajectory $T_{(K)}$ for which the average equals $T_{(K)}$ from $|T_{(K-1)}|+1$ to $|T_{(K)}|$.

The predicted features at time $t+h$ are thus formally given by:

$$\begin{array}{rcll} \hat{X}^{BS}_{t+h_i} & = & \sum_{k=i}^{K} F_i^k \cdot X_{l_i}^k, & i = 1 \dots K \\ & & & h_i = |T_{(i-1)}| + 1 \dots |T_{(i)}| \\ & & & l_i = c_k \dots |T_{(i)}| \end{array} \tag{20}$$

where BS stands for "Bold Strategy" and $|T_{(0)}| = 0$. The value of weight F_i^k is given by:

$$F_i^k = \frac{\exp\left(-\mathcal{D}_{j^*}^{(k)}\right)}{\sum_{k'=i}^{K} \exp\left(-\mathcal{D}_{j^*}^{(k')}\right)}, k = 1 \dots K \tag{21}$$

where $\mathcal{D}_{j^*}^{(k)}$ is the distance between block $B_{j^*}^{(k)}$ and Y_t after sorting distances as detailed in the previous step.

The main advantage of this strategy is to provide long term predictions (as long as the length of the largest selected trajectories), while the main drawback is the possible lack of reliability according to the horizon.

At the end of step 2 (represented in lines 7-13 of Alg. 1), the prediction of observation trajectory $\hat{X}_t$ is known according to the observation block Y_t and to the training dataset $\mathcal{L}$. Note that exponential smoothing using past prediction ($\hat{X}_{t-1}$) can be performed to improve temporal consistency [1] (not used in this paper).

C. Step 3 - Predicted sequence of states

While step 2 computes the predicted sequence of observations, step 3 is concerned by the prediction of future states. Two strategies are proposed:

- Classification of predictions (CPS): the predicted observations given by step 2 are classified into states.
- Direct projection of future state sequence (DPS): an algorithm is developed to project directly the future states using current observations Y_t.

1) Classification of predictions strategy (CPS): This strategy consists in classifying the predicted observations given by step 2 into states. It requires the training of classifiers able to discriminate the different states. For the sake of simplicity, we consider the multiclass classifier called Evidential K-nearest neighbours (EvKNN) [31] which is able to generate a belief mass on the possible states in Ω given an observation. The main feature of this classifier is the possibility to manage belief functions m_t^i provided in the training dataset $\mathcal{L}$ (partially-supervised classification). Moreover, it is a model-free classifier which is a well-suited approach for PHM applications where we generally face imbalanced data [32].

The CPS thus applies the classifier on the predicted observations $\hat{X}_{t+h}, \forall h$ given the training dataset $\mathcal{L}$ and provides a belief mass on the possible states:

$$m_{t+h}^{CPS} \leftarrow \text{EvKNN classifier}(\mathcal{L}, \hat{X}_{t+h}) \tag{22}$$

where CPS stands for "classification of prediction strategy". From this belief mass, a *hard* decision can be made to estimate the state of the current block by using the pignistic transform [10] which computes a probability distribution (suited for decision-making) from the belief mass m_{t+h}^{CPS} (Eq. 8). Repeating this process on blocks composing the predicted observation $\hat{X}_t$, one simply obtains a sequence of states.

2) Direct projection of future state sequence (DPS): The previous strategy for state sequence prediction is efficient if the prediction of observations $\hat{X}_t$ are sufficiently reliable. Indeed, if predictions $\hat{X}_t$ are far from the true trend then the estimated states will be far from the truth too. However, even if in some cases the reliability can be questioned, the state sequence predicted by the first strategy allows the user to have a rough trend of the future states.

In order to avoid the dependency between state sequence prediction to observation prediction, we propose to exploit another strategy that is the *direct projection of future state sequence*. This second strategy draws benefits directly from the training dataset. The main idea is to apply a similar reasoning as for features X_t but now for belief mass m_t.

To go further in details, let consider the set of belief masses for the K nearest neighbours (gray-filled squares in Fig. 4, i.e. $m_t^k, k = 1 \dots K, t = c_k \dots |T_k|$. These K belief masses can be considered as coming from distinct pieces of evidence so that the conjunctive rule of combination $\oplus$ (Eq. 2) can be used:

$$\begin{array}{rcll} \hat{m}_{t+h}^{DPS} & = & \oplus_{k=1}^{K} m_l^k, & l = c_k \dots |T_k| \\ & & & h = 1 \dots \mathcal{P} \end{array} \tag{23}$$

where DPS stands for "direct projection strategy" and $\mathcal{P}$ is given by Eq. 16.

As an illustration, Figure 4 depicts a part of the CPS process.

To apply Eq. 23, two problems have to be solved:

- How to manage different number of BBAs in the fusion process?
- How to manage the conflict appearing in the fusion process?

a) Managing the number of BBAs: In the case where the bold strategy is exploited, we propose to use a *vacuous padding* process as represented in Figure 4. It simply consists in artificially adding new BBAs at the end for all trajectories except the longest one so that all sets of BBAs have equal length. This process enables one to compute easily the fusion (Eq. 23) since the vacuous BBA is the neutral element of the conjunctive rule $\oplus$ (section III-A3).

b) Managing the conflict: The amount of conflict may increase when the BBAs presents contradiction, e.g. when two BBA give important mass to subsets with empty intersection. Moreover, the conflict is potentially higher when the BBAs are focused on singletons.

To decrease the amount of conflict during the fusion process, we propose to use a process called *discounting* [9], [33], which interest is to transfer a *part* of all masses onto the set Ω. By doing so, Dempster's rule generates less conflict. The part is proportional to the weights (Eq. 17), so that the discounting becomes:

$$\begin{array}{rcl} m_{t'}^k(A) & \leftarrow & F^k \times m_{t'}^k(A), \forall A \subset \Omega \\ m_{t'}^k(\Omega) & \leftarrow & (1 - F^k) + F^k \times m_{t'}^k(\Omega) \end{array} \tag{24}$$

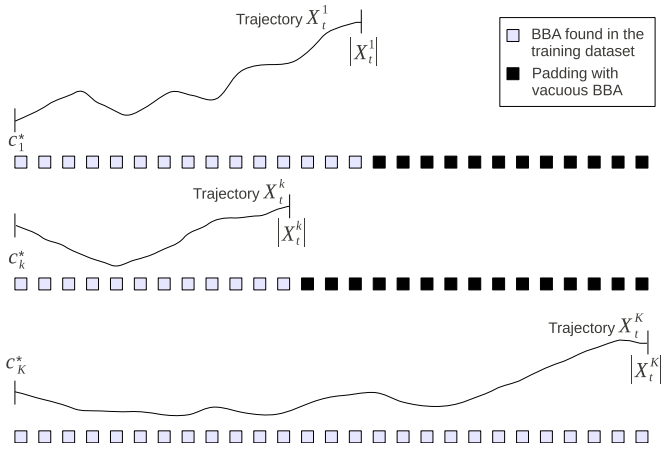


Figure 4. Illustration of the vacuous padding: A vacuous belief mass, representing ignorance about the states, is assumed for all cases where the BBA is not available (black-filled squares), e.g. in the bold strategy.

The highest the weight, the less the discount, meaning that the related BBA is trusted. Once the BBAs have been discounted, the estimated belief mass at time t in DPS is given by Eq. 23.

Figure 5 is an illustration of a particular DPS proposal where the belief masses have only categorical assignments (i.e. all masses are focused on only one singleton at each time-step). The state is precisely known for each data in the training set and therefore, the states at each time step of the prediction $(t+h)$ can be estimated by direct projection of their values using a similar process as for the CPS (Figure 2). The dashed line represents the sequence obtained after fusion (Step 4) and the continuous bold line depicts the sequence obtained after fusion at the current iteration of the algorithm. The median value and the standard deviation of time instants of transitions from states q to r within the set of sequences computed at each iteration can be computed to locate transitions. In this case, the direct propagation seems like a projection of possible sequences of states.

Step 3 is represented in lines 14-15 of Alg. 1.

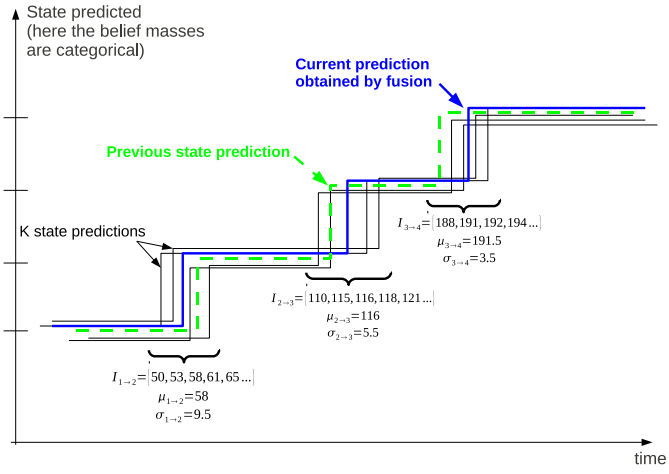


Figure 5. Illustration of the state prediction process for the case where the belief masses distribution are focused only on a singletons.

D. Step 4 - Remaining Useful Life (RUL) estimation

Step 4 is the final step of the EVIPRO-KNN algorithm. The goal of this step is to estimate the transition to a potential "faulty" state indicating a potential damage of the system monitored.

1) CPS and DPS fusion: To draw benefits from both CPS and DPS approaches, the BBAs m_{t+h}^{CPS} (Eq. 22) and m_{t+h}^{DPS} (Eq. 23) are combined and the resulting BBA is converted into a probability distribution from which a decision can be made [34]. Dempster's rule is not adapted for the fusion of CPS and DPS's BBAs because m_{t+h}^{CPS} and m_{t+h}^{DPS} can not be considered as coming from distinct bodies of evidence. Indeed:

- CPS is a classification of predictions resulting from the weighted combination of continuous predictions,
- DPS generates belief masses discounted by the weights,

and therefore, both approaches depend on the weights. Moreover, both rely on the BBAs in the training dataset $\mathcal{L}$.

Thus, the fusion may be performed using the cautious rule (Eq. 3). The main disadvantage of the cautious rule comes from the fact that the neutral element is not always the vacuous BBA [26]. For EVIPRO-KNN, this can be a problem in practice if the belief masses are not available in the training dataset (for DPS). Indeed, in this case, the fusion process between CPS and DPS does not lead to CPS. However, for particular BBA called separable BBA [26], [35], the neutral element *is* the vacuous BBA, thus leading to an expected behavior. Besides, when using the EvKNN classifier as proposed in this paper, the BBAs generated *are* separable, and this is also the case for other commonly used classifier [24], [26].

Conclusively, the fusion process relies on the cautious rule and leads to a predicted BBA (Eq. 5, 4 and 3, followed by 6 and 7):

$$\hat{m}_{t+h} = m_{t+h}^{CPS} \circledwedge m_{t+h}^{DPS} \tag{25}$$

from which a decison concerning the state at time $t+h$ can be made using Eq. 8. The result of this phase is the estimation of a sequence of states $\hat{\omega}_{t+h}$. An illustration of such sequences is depicted in Figure 5.

2) RUL estimates: Let now consider this sequence of states but also all previous predicted sequences (Figure 5). Based on this set of sequences, we propose to compute some expected sufficient statistics of time instants of transition between states. Since each sequence is composed of possible transitions between some states q and r, the set of time instants of transitions between both states is:

$$I_{q\to r} = \{t : \hat{\omega}_{t-1} = q \text{ and } \hat{\omega}_t = r\} \tag{26}$$

To estimate the Remaining Useful Life (RUL) of the system, it is sufficient to determine the location of the critical transition from state $q =$ "degrading state" to state $r = q+1 =$ "fault state":

$$\text{transition } q \to r \text{ critical } \Rightarrow RUL = \mu_{q,r} - t \tag{27}$$

where $\mu_{q,r}$ is the estimated time from t to the transition between the degrading state q and the faulty state r that can be computed by a median. It can be associated to a dispersion

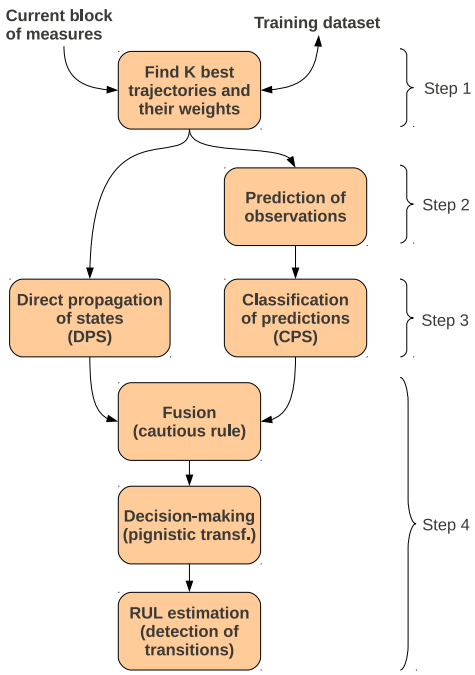


Figure 6. The sequence of operations involved in EVIPRO-KNN.

$\sigma_{q\to r}$ that we computed using the interquartile range:

$$\begin{aligned} \mu_{q\to r} &= \text{median } (I_{q\to r}) \\ \sigma_{q\to r} &= Q_3 - Q_1 \end{aligned} \tag{28}$$

where Q_i is the i-th quartile and $n_I = |I_{q\to r}|$ is the number of elements in the set of time instants of transition $I_{q\to r}$. The step 4 is represented in lines 16-19 of Alg. 1.

Therefore, both methods for sequence prediction, CPS (classification) and DPS (direct projection), assume that each trajectory in the training dataset is made of *at least* two states, say "normal state" and "abnormal state", and knowledge on these states can be uncertain and imprecise and represented by belief functions.

The final algorithm is given in Alg. 1, and the plot chart of the whole algorithm is depicted in Figure 6.

Algorithm 1 EVIPRO-KNN

Require: Training dataset $\mathcal{L}$ {Set of trajectories with observations and belief masses, Eq. 9 and 10}
Require: Current window Y_t {$W \in [20\ 40]$, overlap: $W/2$}
Require: Number of nearest neighbours {$K = 3$}
Require: Labels of critical states {if $S = 4$, then ω_3 and ω_4}
Require: Averaging strategy {Cautious or Bold}
Require: A classifier of states {For CPS}

Ensure: Prediction of observations $\hat{X}$
Ensure: Remaining Useful Life (RUL) estimation

{*Step 1 - Find nearest neighbours*}
1: **for all** all trajectories i **do**
2: Find the closest block to Y_t {Eq. 11 and Eq. 12}
3: Store distances {Eq. 13}
4: **end for**
5: Keep the K best trajectories {Eq. 14}
6: Compute weights {Eq. 17}
{*Step 2 - Compute prediction of observations*}
7: **if** T_k have same length $\forall k = 1 \dots K$ **then**
8: $\hat{X} \leftarrow$ Apply Eq. 15
9: **else if** Cautious strategy **then**
10: $\hat{X} \leftarrow$ Apply Eq. 19 (and Eq. 18, 17)
11: **else if** Bold strategy **then**
12: $\hat{X} \leftarrow$ Apply Eq. 20 (and Eq. 21)
13: **end if**
{*Step 3 - Prediction of states*}
14: $m_{t+h}^{CPS} \leftarrow$ Apply the classifier {Step 3-1, Eq. 22}
15: $m_{t+h}^{DPS} \leftarrow$ Apply direct projection {Step 3-2, Eq. 23, 24}
{*Step 4 - RUL estimation*}
16: $\hat{m}_{t+h} \leftarrow$ Eq. 25 {Fusion of m_{t+h}^{CPS} and m_{t+h}^{DPS}}
17: $I_{q\to r} \leftarrow$ Find transitions between critical states and store time instants {Eq. 26, 8}
18: Find median and standard deviation {Eq. 28}
19: RUL $\leftarrow$ Apply Eq. 27

V. Experiments

The goal is to illustrate the capability of EVIPRO-KNN algorithm to provide reliable health assessment and long term predictions.

We considered the challenge dataset concerning diagnostic and prognostics of machine faults from the first Int. Conf. on Prognostics and Health Management [36]. The dataset is a multiple multivariate time-series (26 variables) with sensor noise. Each time series was from a different engine of the same fleet and each engine started with different degrees of initial wear and manufacturing variation unknown to the user and considered normal. The engine was operating normally at the start and developed a fault at some point. The fault grew in magnitude until system failure.

In this paper, we used the first experiment found in the text file *train_FD001.txt* that is composed of 100 time-series and we considered only 5 features among 26 (columns $7, 8, 9, 13, 16$ in the text file, as proposed by [37] for the first 5 features[2]).

Figure 7(a) pictorially described the evolution of the first feature for the 100 training data. This figure emphasizes the difficulty to build statistical models based on duration since a high variability is present.

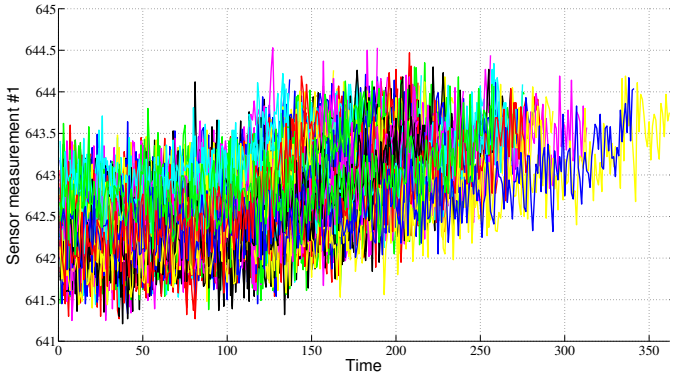
A. First test: segmented data with 4 states

The goal of this first set of experiments is to evaluate the proposed algorithm. We particular aim at:

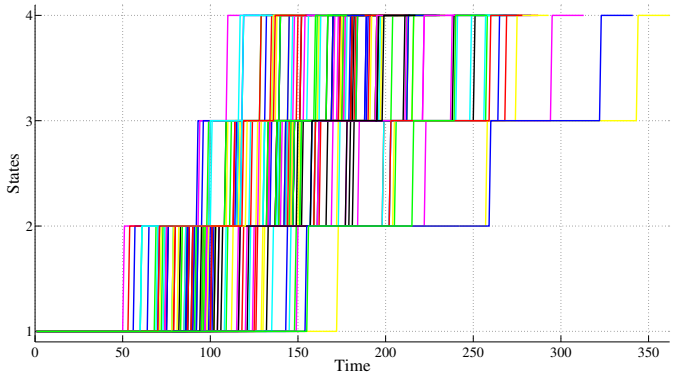
- Comparing the bold and the cautious strategies.
- Studying the sensitivity on K (number of neighbours) and W (window size).

1) Protocol:

[2]In [37], the authors proposed to use sensor measurements 2, 3, 4, 7, 11, 12, 15 corresponding to columns 7, 8, 9, 12, 16, 17, 20, and we believe that 12 should be replaced by 13 because 12 is almost flat and does not bring information.



(a) First feature for 100 training data.



(b) Corresponding state sequences.

Figure 7. (Top) Evolution of the first feature for all trajectories in the training dataset, and (Bottom) evolution of the state sequences in the training dataset (after decision-making based on the belief masses).

a) Presentation of the data: In the first test, each time-series was manually segmented into 4 functioning states[3]: normal mode (ω_1, label state 1), transition mode (ω_2, label state 2), degrading mode (ω_3, label state 3) and faulty mode (ω_4, label state 4). Thus $\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4\}$ is the set of states.

An illustration of this manual segmentation is depicted in Figure 7(b). This figure clearly emphasizes that approaches based on state duration [38] are not reliable here. Variability concerning time instant of transition from one state to another points out that the first states are generally similar (all normal modes are almost the same) while faulty modes are very different (a fault state can appear at time $t = 120$ or at $t = 280$). This variability is generally encountered in many real systems where the degradation is mainly mechanical with many complex interactions between the components.

b) Meaning of belief functions: Since the transitions between modes are not known precisely, belief functions are used as follows: let $t_{i\to j}$ denote the time instant of a transition between state ω_i (before) and ω_j (after), both in Ω, then the belief masses around this time instant are simply set to $m_t(\{\omega_i, \omega_j\}) = 1, \forall t \in [t_{i\to j} - 5, t_{i\to j} + 5]$. With such belief masses, doubt between both states ω_i and ω_j is made explicit in the transitions.

c) Building the classifier: For the CPS process, we used the Evidential K-nearest neighbours (EvKNN). This algorithm requires a training dataset made of feature vectors (here made of data points of trajectories) and of belief masses (provided in the training dataset), where a belief mass reflects how the corresponding feature vector belongs to each subset of states.

d) Evaluation process: A leave-one-out evaluation was performed: 99 time-series were used as the training dataset and the remaining time-series was used as the testing data. The testing data is analysed block by block by EVIPRO-KNN algorithm until the end of the time-series. Given a block, the algorithm may propose a value for the Remaining Useful Life (RUL). The estimate of the RUL can be either early (before than expected) or late, but early predictions are generally preferred [39] (Figure 8). To assess the predictions, we first define the prediction error at a given time by [39]:

$$E = \text{actual ToF} - \text{predicted ToF} \tag{29}$$

where ToF stands for Time-of-Failure. In [39], the authors also distinguished between:

- False Negatives (FN) cases corresponding to late predictions such as $E < -t_{FN}$ where t_{FN} is a user-defined FN threshold:

$$FN = \begin{cases} 1 & \text{if } E < -t_{FN} \\ 0 & \text{otherwise} \end{cases} \tag{30}$$

- False Positives (FP) cases corresponding to early predictions such as $E > t_{FP}$ where t_{FP} is a user-defined FP threshold:

$$FP = \begin{cases} 1 & \text{if } E > t_{FP} \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

The meaning of thresholds is represented in Figure 8. In the sequel interval $[t_{FN}, t_{FP}]$ is denoted $\mathcal{I}$.

We considered $\mathcal{I} = [10, +13]$ which is a severe condition on the prediction compared to the literature [3], [39]. These conditions are similar to the ones considered in the PHM'08 data challenge [37] as well as in the PHM'12 data challenge [40] except that, in both papers, a particular parametrized function was used which decreases with report to the bounds of $\mathcal{I}$.

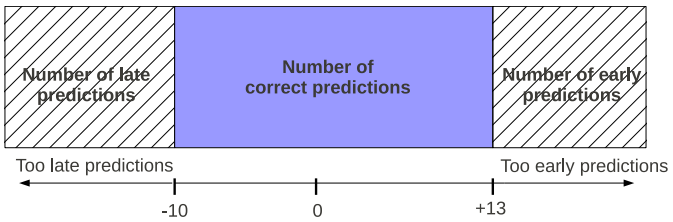


Figure 8. Metric of performance assessment, here $\mathcal{I} = [-10, +13]$.

The performance of the algorithm is quantified by the percentage of predictions provided at the so-called critical t_c and falling in $\mathcal{I}$. The critical time sets the difficulty of the prediction task by specifying how far the prediction has to start [39]. For example, given a trajectory T with a total length equal to $|T|$ (on which the algorithm is applied), setting $t_c = 20$ means that the RUL estimated at $|T| - 20$ is evaluated.

[3]The segmentation is available at http://www.femto-st.fr/~emmanuel.ramasso/PEPS_INSIS_2011_PHM_by_belief_functions.html.

Therefore the greater the value of t_c the more difficult is the task. However, t_c is generally set arbitrarily.

Since the variability of the length of the trajectories in the dataset is important (as shown in Fig. 7(b)), we proposed to set $t_c = |T|/4$, meaning that the value of the critical time t_c is adapted according the trajectory's length (e.g. if $|T| = 240$ then $t_c = 60$). The distribution of the critical times will be given in the sequel (Fig. 10) and we will show that $t_c > 22$ which corresponds to mid/long-term predictions.

This way of evaluating the predictions is of practical interest because, in a real-world application, one does not know *a priori* the length of the current trajectory (which is under analysis). For example, for the second application (next section), only pieces of trajectories with different length are available for testing, and nothing is known about the associated system's status. In this second application, the evaluation of the RUL proposals made by the EVIPRO-KNN algorithm will thus inherently consist in setting t_c equal to the length of the current piece of trajectory.

2) Sensitivity of K and W: Figure 9 depicts the performance of EVIPRO-KNN with report to $K = [1\ 3\ 5\ 9]$ (number of neighbours) and $W = [10\ 20\ 30\ 40]$ (window's size), with the bold strategy. The algorithm is robust to both parameters, with a performance close to 92% for $K > 9, W > 30$. For low values of W, the performances are the lowest (around 68%), however, these cases correspond to greedy ones since the algorithm requires more iterations and, therefore, are not practically interesting. Increasing W makes EVIPRO-KNN less sensitive to K (e.g. $W = 40$) because the window's size provides the best choice of the nearest neighbours. This effect is emphasized when studying the effect of the number of neighbours. Indeed, low values of W (say < 25) requires low values for K (say < 3). For high values of W (say ≥ 25), $K = 3$ or $K = 5$ yields satisfying performances. Practically, W should not be too high to avoid missing changes in the trajectories, and therefore its settings is application-dependent. In the considered application, the fastest changes took between 20 and 30 time-units as illustrated in Figure 7(a) (see the shortest trajectories).

As mentioned previously, the distribution of the critical times is given in Figure 10 which shows that $t_c > 22$. These critical times can be considered as difficult ones compared to the literature on PHM.

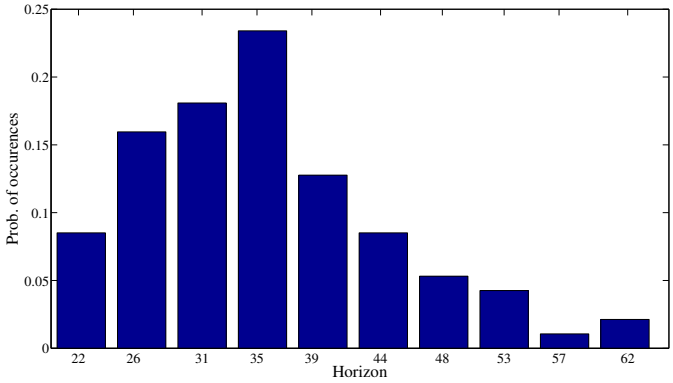


Figure 10. Distribution of the critical times.

3) Cautious or bold strategy?: Figure 11 pictorially represents the evolution of the performance of EVIPRO-KNN with report to $K = [1\ 3\ 5\ 9]$ and $W = [10\ 20\ 30\ 40]$, with the cautious strategy. Remembering that both strategies differ from the way the farthest predictions are computed, the small differences with Fig. 9 can be explained by the relatively small values taken by the critical times. Even though these critical times correspond to quite difficult cases (mid-term predictions), they do not enable one to discriminate the prediction capability of both strategies.

To have a better illustration of which strategy one may choose, let consider the distribution of the time-instant of the *five* farthest and correct predictions (falling in $\mathcal{I}$) for each testing data. A similar idea was suggested in [39] (where the authors considered only *the* farthest prediction) and enables one to assess very-long term predictions. Figures 12(a) and 12(b) represent the distribution of these farthest critical times for $W = 30$ and $K = 3$. The bold strategy provides the shortest critical times compared to the cautious strategy which was not expected. And the differences are significant: one

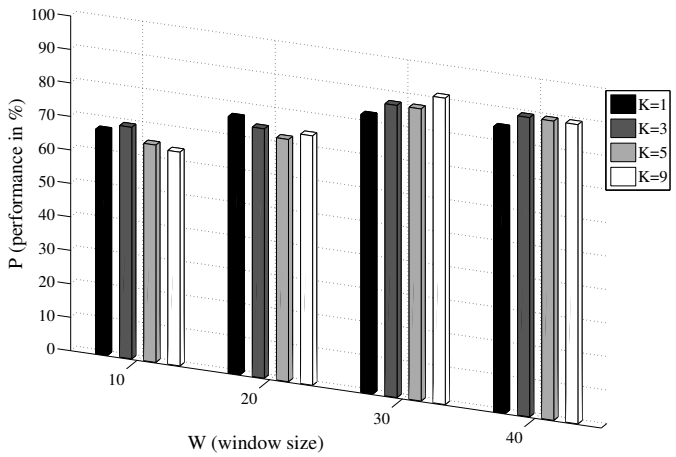


Figure 9. Performance with report to K and W for mid-term prediction for the bold strategy.

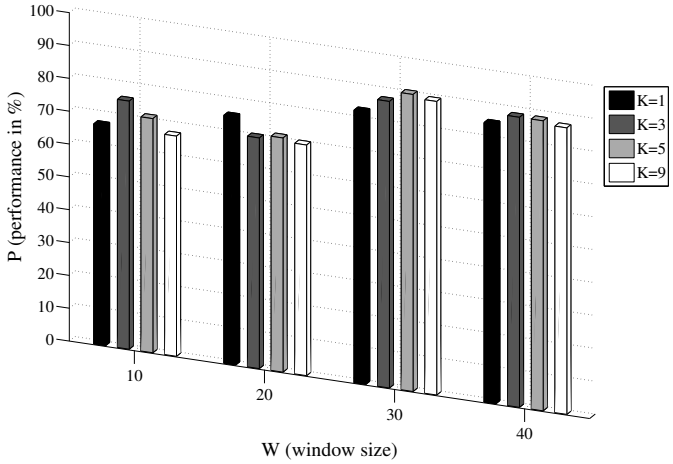
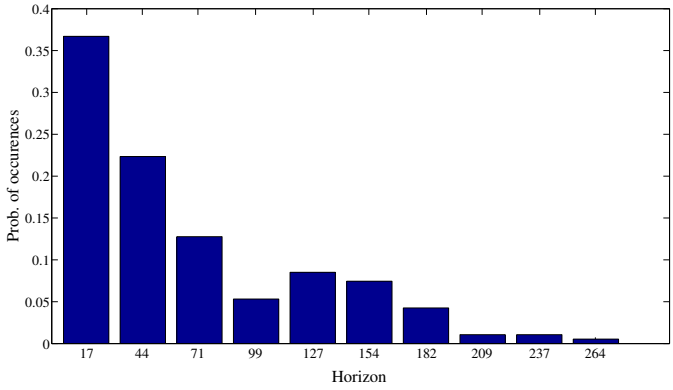
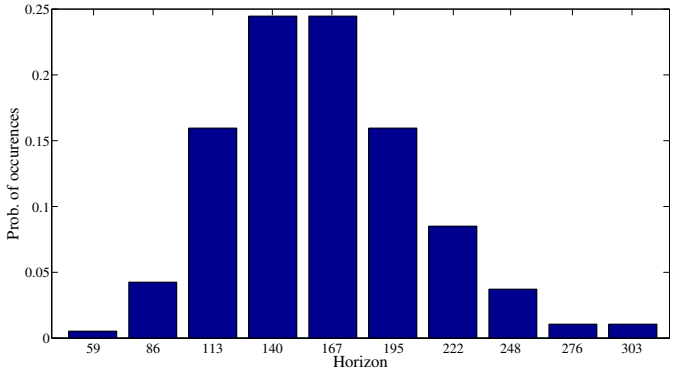


Figure 11. Performance with report to K and W for mid-term prediction for the cautious strategy.

may expect an horizon less than 50 (most probable cases) for efficient predictions by the bold strategy, whereas an horizon between 100 and 200 can be expected using the cautious strategy. Therefore, the cautious strategy is the best one on this dataset. An explanation can be found in the way the predictions are computed in the bold strategy (Eq. 20) which induces some discontinuities in the predictions. This side-effect is an inherent behavior of the proposed approach.



(a) Bold strategy



(b) Cautious strategy

Figure 12. Expected horizon for long-term prediction with the bold and cautious strategies.

4) Error between true and estimated rul: Figure 13 illustrates the evolution of the differences at each time-step between the estimated RUL and the real RUL for $W = 30$ and $K = 3$. A convergence towards the real value is observed as expected. For this example, a good estimate of the RUL (in interval $[-10, 13]$) is obtained at $t = 90$, so 180 time-units in advance.

5) Error between truth and predictions of observations: Figure 14 depicts both the mean-squared error between the prediction of observations and the ground truth (subfigure 1) and the length of the prediction (horizon) proposed by the algorithm (subfigure 2). As expected, both quantities decrease as the end of the time-series approaches (with $W = 30$ and $K = 3$).

6) Online segmentation: An interesting characteristic of EVIPRO-KNN algorithm is the possibility to predict transitions of all states. For instance, in Figure 15, an online segmentation of the signal is proposed at a critical time

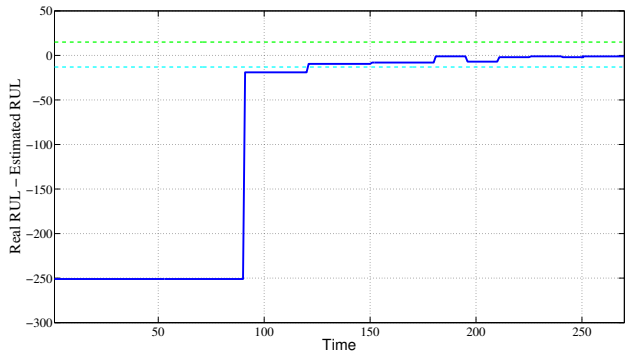


Figure 13. Differences between estimated RUL and real RUL for each window.

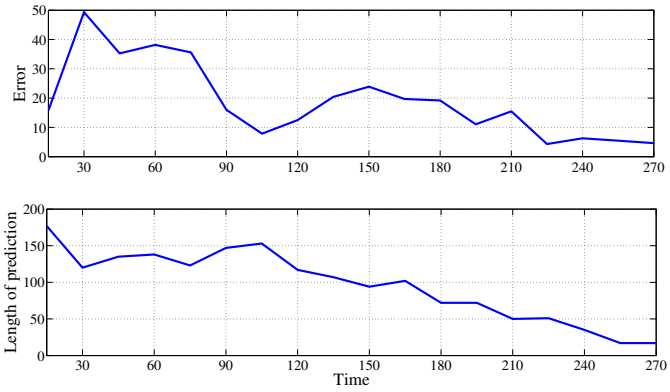


Figure 14. Error between predictions and real observations (top) and the length of the predictions (bottom).

$t_c = 40$ for $W = 30$ and $K = 3$. The signal takes continuous values but here it was rescaled for the sake of interpretation. This segmentation can be useful to detect the future state sequences.

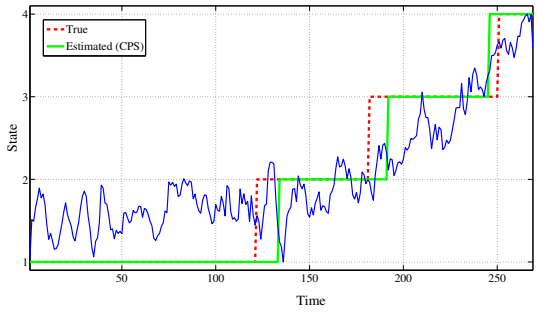


Figure 15. In dashed-line is represented the true segmentation of the signal, the segmentation proposed by the algorithm appears with the continuous line (dicrete values).

7) Trajectories hits: Figure 16 represents the index of the trajectory chosen for each block (first neighbour) for $W = 30$ and $K = 3$. This figure allows the user to assess the number of times a trajectory was chosen for the prediction. For example trajectory $i = 82$ is used for blocks $4, 5$ and 6. In this application, one can observe that different trajectories are used, demonstrating that the training dataset contains

complementary information. In practice, these information can be useful for condensing the training dataset by removing the trajectories which are not often used.

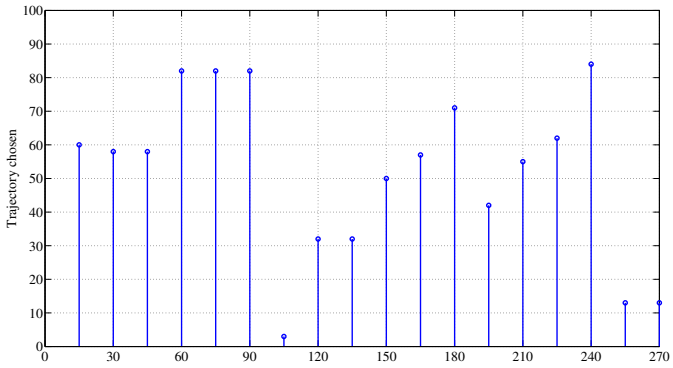


Figure 16. Trajectory hits describing which trajectory is chosen at each window.

B. PHM 2008 dataset: Pieces of trajectories

1) Presentation of the data: For the second test, we considered again the challenge dataset of machine faults prediction from the first Int. Conf. on Prognostics and Health Management [36]. The training dataset is the same as in the previous set of experiments (100 trajectories) but the testing dataset (available on the website of the conference in the text file *test_FD001.txt* and composed of 100 trajectories) is composed only of pieces of trajectories (the remaining is unknown). Besides, the length of the testing trajectories is not the same for the whole dataset (Fig. 17).

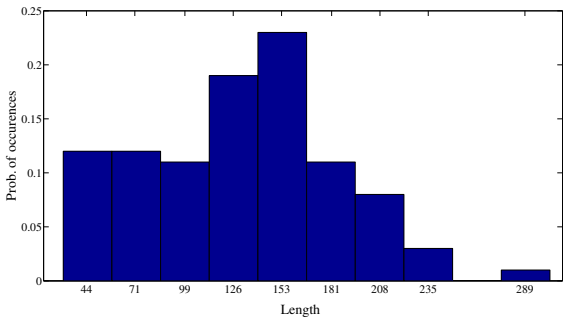


Figure 17. Distribution of the length of the trajectories in the second dataset.

Given one of these 100 pieces of trajectory, the goal is to predict the Remaining Useful Life (RUL). The predicted RUL are compared with the true RUL provided in the text file *rul_FD001.txt*. The RULs are spread in $[14, 138]$ as depicted in Figure 18, with more than the half greater than 80 (very long-term prediction).

2) Belief functions, classifier and EVIPRO-KNN's settings: The belief masses were the same as in the previous section. Besides, the same procedure as in the previous test is applied to build the classifier. Concerning the EVIPRO-KNN algorithm, and regarding the previous results, we selected $K = 3$, $W = 30$ and the cautious strategy.

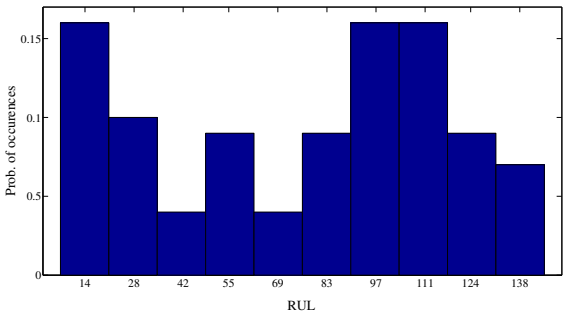


Figure 18. Distribution of the RULs in the second dataset.

3) Evaluation process: To assess the results, we compare the estimated RUL with the RUL provided in the file *rul_FD001.txt* using interval $\mathcal{I} = [-10, 13]$. For a given tested trajectory, the critical time t_c is given by the length of this trajectory.

4) Results: Figure 19 depicts the histogram of errors made by the EVIPRO-KNN algorithm on the second dataset. The number of satisfying cases, i.e. falling in $\mathcal{I} = [-10 \;\; 13]$, is about 53%. The amount of late predictions (corresponding to negative errors) is about 11% and 36% of early predictions. Early predictions are practically preferred to late ones because the algorithm does not miss potential breakdowns.

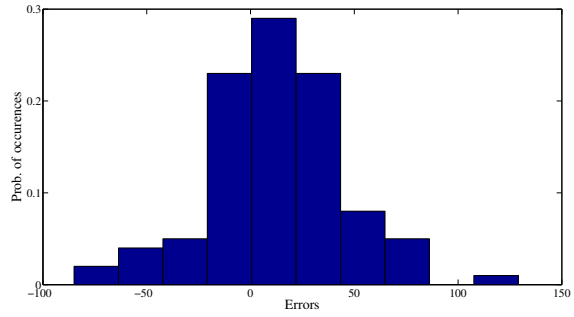


Figure 19. Distribution of errors of the RUL's estimates for the second dataset.

VI. Conclusion

EVIPRO-KNN is an online algorithm for prognostics and health detection that takes as input an observation (test instance) in the form of a piece of a trajectory and then generates the prediction as the weighted sum of the K-nearest trajectories found in the training dataset. Compared to previous approaches, EVIPRO-KNN is based on belief functions which allow the user to manage labels that can possibly be assigned imprecisely to the training data.

The possibility to manage prior information on possible states in the training dataset is one of the main advantages of EVIPRO-KNN. Indeed, in most papers, only two states are considered: normal and faulty, while in many real cases, more states have to be considered. The other main asset of EVIPRO-KNN is the possibility to predict sequence of continuous observations jointly with discrete states. These sequences

allow the user to have access to the online segmentation of the current observed data and may be practically useful. The joint prediction is made by two strategies which finally provide an estimate of the Remaining Useful Life: the classification of predictions (CPS) and the direct projection of future state sequence (DPS).

To increase accuracy of the predictions, in particular made by CPS, we are currently studying classifiers and fusion rules [41]. However, to perform online detection and prediction as considered in this paper, it is required to develop online classifiers and online fusion tools.

ACKNOWLEDGMENT

This work is supported by a PEPS-INSIS-2011 grant from the French National Center for Scientific Research (CNRS) under the administrative authority of France's Ministry of Research. We also thank the anonymous reviewers for their helpful comments that greatly improved the paper.

REFERENCES

[1] P. Bonissone, A. Varma, and K. Aggour, "A fuzzy instance-based model for predicting expected life: A locomotive application," in *IEEE Int. Conf. on Computational Intelligence for Measurement Systems and Applications*, 2005, pp. 20–25.

[2] A. Saxena, B. Wu, and G. Vachtsevanos, "Integrated diagnosis and prognosis architecture for fleet vehicles using dynamic case-based reasoning," in *Autotestcon*, 2005, pp. 96–102.

[3] T. Wang, "Trajectory similarity based prediction for remaining useful life estimation," Ph.D. dissertation, University of Cincinnati, 2010.

[4] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, August 2006.

[5] K. P. Murphy, "Dynamic Bayesian networks: Representation, inference and learning," Ph.D. dissertation, UC Berkeley, 2002.

[6] E. Ramasso and R. Gouriveau, "Prognostics in switching systems: Evidential Markovian classification of real-time neuro-fuzzy predictions," in *IEEE Int. Conf. on Prognostics and System Health Management*, Macau, China, 2010, pp. 1–10.

[7] M. El-Koujok, R. Gouriveau, and N. Zerhouni, "Reducing arbitrary choices in model building for prognostics: An approach by applying parsimony principle on an evolving neuro-fuzzy system," *Microelectronics reliability*, vol. 51, pp. 310–320, 2011.

[8] A. Dempster, "Upper and lower probabilities induced by multiple valued mappings," *Annals of Mathematical Statistics*, vol. 38, pp. 325–339, 1967.

[9] G. Shafer, *A mathematical theory of Evidence*. Princeton University Press, Princeton, NJ, 1976.

[10] P. Smets and R. Kennes, "The Transferable Belief Model," *Artificial Intelligence*, vol. 66, no. 2, pp. 191–234, 1994.

[11] P. Smets, "What is Dempster-Shafer's model ?" in *Advances in the Dempster-Shafer Theory of Evidence*, I. R. Yager, M. Fedrizzi, and J. Kacprzyk, Eds. J.Wiley & Sons, 1994, pp. 5–34.

[12] L. Serir, E. Ramasso, and N. Zerhouni, "Time-sliced temporal evidential networks: the case of evidential hmm with application to dynamical system analysis," in *IEEE Int. Conf. on Prognostics and Health Management*, 2011, pp. 1–10.

[13] E. Come, L. Oukhellou, T. Denoeux, and P. Aknin, "Learning from partially supervised data using mixture models and belief functions," *Pattern Recognition*, vol. 42, no. 3, pp. 334–348, 2009.

[14] O. E. Dragomir, R. Gouriveau, N. Zerhouni, and R. Dragomir, "Framework for a distributed and hybrid prognostic system," in *4th IFAC Conf. on Management and Control of Production and Logistics*, 2007.

[15] H. Papadopoulos, V. Vovk, and A. Gammerman, "Regression conformal prediction with nearest neighbours," *Journal of Artificial Intelligence Research*, vol. 40, pp. 815–840, 2011.

[16] G. Nakhaeizadeh, "Learning prediction of time series - a theoretical and empirical comparison of cbr with some other approaches," in *First European Workshop on Topics in Case-Based Reasoning*, 1993, pp. 65 – 76.

[17] D. Ruta, D. Nauck, and B. Azvine, "K-nearest sequence method and its application to churn prediction," in *Intelligent Data Engineering and Automated Learning*, 2006, pp. 207–215.

[18] Q. Yu, A. Sorjamaa, Y. Miche, and E. Severin, "A methodology for time series prediction in finance," in *European Symposium on Time Series Prediction*, 2008, pp. 285–293.

[19] Q. Yu, A. Lendasse, and E. Severin, "Ensemble KNNs for bankruptcy prediction," in *Int. Conf. on Computing in Economics and Finance*, 2009.

[20] S. Zhang, W. Jank, and G. Shmueli, "Real-time forecasting of online auctions via functional k-nearest neighbors," *International Journal of Forecasting*, vol. 26, pp. 666 – 683, 2010.

[21] M. Dong and D. He, "A segmental hidden semi-markov model (hsmm) -based diagnostics and prognostics framework and methodology," *Mech. Syst. Signal Processing*, vol. 21, pp. 2248–2266, 2007.

[22] O. Ayad, M. S. Mouchaweh, and P. Billaudel, "Switched hybrid dynamic systems identification based on pattern recognition approach," in *IEEE International Conference on Fuzzy Systems*, 2010, pp. 1–7.

[23] K. Goebel, N. Eklund, and P. Bonanni, "Fusing competing prediction algorithms for prognostics," in *IEEE Aerospace Conference*, 2006.

[24] T. Denoeux and P. Smets, "Classification using belief functions: The relationship between the case-based and model-based approaches," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 36, no. 6, pp. 1395–1406, 2006.

[25] L. Serir, E. Ramasso, and N. Zerhouni, "Evidential evolving Gustafson-Kessel algorithm for data streams partitioning using belief functions," in *The 11th European Conf. on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, vol. LNAI 6717, 2011.

[26] T. Denoeux, "Conjunctive and disjunctive combination of belief functions induced by non distinct bodies of evidence," *Artificial Intelligence*, vol. 172, pp. 234–264, 2008.

[27] A. Dempster, "A generalization of Bayesian inference," *Journal of the Royal Statistical Society*, vol. 30, pp. 205–247, 1968.

[28] P. Smets, "Analyzing the combination of conflicting belief functions," *Information Fusion*, vol. 8, no. 4, pp. 387–412, October 2005.

[29] R. Kennes and P. Smets, "Fast algorithms for dempster-shafer theory," in *Uncertainty in Knowledge Bases*, L. Z. B. Bouchon-Meunier, R.R. Yager, Ed., vol. Lecture Notes in Computer Science 521. Springer-Verlag, Berlin, 1991, pp. 14–23.

[30] J. Denker and Y. Lecun, "Transforming neural-net output levels to probability distributions," in *Advances in Neural Information Processing Systems 3*. Morgan Kaufmann, 1991, pp. 853–859.

[31] T. Denoeux, "A k-nearest neighbor classification rule based on Dempster-Shafer theory," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 5, pp. 804–813, 1995.

[32] H. He and E. Garcia, "Learning from imbalanced data," *IEEE Trans. on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[33] P. Smets, "Beliefs functions: The Disjunctive Rule of Combination and the Generalized Bayesian Theorem," *Int. Jour. of Approximate Reasoning*, vol. 9, pp. 1–35, 1993.

[34] ——, "Decision making in the TBM: The necessity of the pignistic transformation," *Int. Jour. of Approximate Reasoning*, vol. 38, pp. 133–147, 2005.

[35] ——, "The canonical decomposition of a weighted belief," in *In Int. Joint Conf. on Artificial Intelligence*. San Mateo, Ca: Morgan Kaufman, 1995, pp. 1896–1901.

[36] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Int. Conf. on Prognostics and Health Management*, Denver, CO, USA, 2008, pp. 1–9.

[37] T. Wang, Y. Jianbo, D. Siegel, and J. Lee, "A similarity-based prognostics approach for remaining useful life estimation of engineered systems," in *IEEE Int. Conf. on Prognostics and Health Management*, 2008.

[38] M. Dong and D. He, "A segmental hidden semi-markov model (HSMM)-based diagnostics and prognostics framework and methodology," *Mechanical Systems and Signal Processing*, vol. 21, pp. 2248–2266, 2007.

[39] K. Goebel and P. Bonissone, "Prognostic information fusion for constant load systems," in *7th Annual Conference on Information Fusion*, 2003, pp. 1247–1255.

[40] P. Nectoux, R. Gouriveau, K. Medjaher, E. Ramasso, B. Morello, N. Zerhouni, and C. Varnier, "PRONOSTIA: An experimental platform for bearings accelerated life test," in *IEEE Int. Conf. on PHM*, Denver, CO, USA, 2012, submitted.

[41] E. Ramasso and S. Jullien, "Parameter identification in Choquet integral by the Kullback-Leibler divergence on continuous densities with application to classification fusion," in *European Society for Fuzzy Logic and Technology*, ser. Advances in Intelligent Systems Research, Aix-Les-Bains, France, 2011, pp. 132–139.